Amendments to the Specification

Please replace the paragraph beginning at page 5, line 1, with the following rewritten paragraph:

> While executing other code or instructions (e.g., 116), the thread desires to utilize a shared resource 104. The thread, desiring to utilize the shared resource 104, calls 126 the wait resource 112. The wait resource 112 has two steps. First, the wait resource calls an exchange resource 120, that exchanges the threads local variable 124, with the global variable 114. Preferably, the exchange resource is executed atomically without interruption. Second, the wait resource 112 jumps to or calls the address contained in the local variable 124.

Please replace the paragraph beginning at page 2, line 4, with the following rewritten paragraph:

> In one example, a global variable contains a pointer to either a protected resource, or a wait resource. The global variable is initialized to point to the protected resource. Plural threads (or processes), attempt to obtain access to the protected resource, by jumping to an address contained in a local variable associated with the respective thread. The local variable for a thread is initialized to point to the wait resource. When a thread wants to enter the protected resource, it jumps to the ~~wiat~~ wait resource. The wait resource exchanges the local variable with the global variable and then jumps to (or calls) the address in the local variable (i.e., the address received in the exchange). Since the global variable is initialized with the pointer to the protected resource, the first thread exchanging variables, obtains the pointer to the protected resource. Thus, by jumping to the address pointed to by the local variable, the thread has entered the protected resource. No other thread (or process) can enter the protected resource while the thread holds the pointer to the protected resource in its' local variable.

Please replace the paragraph beginning at page 4, line 24, with the following rewritten paragraph:

When a thread is created for execution, data specific to the thread is loaded into memory 106, and made available to the executing thread. For example, the thread includes a local variable 124 that is initialized with a pointer 126 to a wait resource 112 ~~122~~.